

AD-A065 644

VIRGINIA POLYTECHNIC INST AND STATE UNIV BLACKSBURG --ETC F/G 9/2
TRANSFER OF APL WORKSPACES: A USEFUL SOLUTION.(U)
SEP 78 K RAUTENKRANZ, R J ORGASS

AFOSR-79-0021

UNCLASSIFIED

CS78006-T

AFOSR-TR-79-0238

NL

1 OF 1
AD
A065644



END
DATE
FILMED
5-79
DDC

AFOSR-TR- 79 - 0238

LEVEL

4

AD A0 65644

THIS DOCUMENT IS BEST QUALITY PRACTICES.
THE COPY FURNISHED TO DDC CONTAINED A
SIGNIFICANT NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

Computer Science Department



DDC FILE COPY



Virginia Polytechnic Institute
and State University
Blacksburg, Virginia 24061

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This document has been reviewed and is
approved for public release in accordance with AFM 190-12 (7b).
Distribution is unlimited.
A. D. BLOFF
Technical Information Officer

Approved for public release;
distribution unlimited.

9 03 12 003

DISCLAIMER NOTICE

**THIS DOCUMENT IS BEST QUALITY
PRACTICABLE. THE COPY FURNISHED
TO DDC CONTAINED A SIGNIFICANT
NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.**

9 14
Technical Report, CS78006-T

6
TRANSFER OF APL WORKSPACES:
A USEFUL SOLUTION

by

10 Karl/Rautenkranz

and
Richard J./Orgass

DDC
RECEIVED
MAR 14 1979
RECEIVED
C

16 2304 17 A2
Department of Computer Science

College of Arts and Sciences
Virginia Polytechnic Institute & State University
Blacksburg, Virginia 24061

18 AFOSR

19 TR-79-0238

12 300.

11 September 1978

15 AFOSR-79-0021

* Research Sponsored by the Air Force Office of Scientific Research,
Air Force Systems Command, Under Grant No. AFOSR-78-3499 to the
University of Arizona. The United States Government is authorized
to reproduce and distribute reprints for Governmental purposes
notwithstanding any copyright notation hereon.
Grant no. AFOSR-79-0021

† Current address: Trent University, Peterboro, Ontario K9J 7B8, Canada.

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
NOTICE OF TRANSMISSION TO DDC

This technical report has been reviewed and is
approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.

A. D. BAILEY
Technical Information Officer

411 098

LB

Copyright, 1978

by

Karl Rautenkranz

and

Richard J. Orgass

General permission to republish, but not for profit, all or part of this report is granted, provided that the copyright notice is given and that reference is made to the publication (Technical Report No. CS78006-T, Department of Computer Science, Virginia Polytechnic Institute and State University, to its date of issue (September, 1978) and to the fact that reprinting privileges were granted by the author.

- A -

ABSTRACT

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY NOTES	
DATE and/or	
A 23	E.A.

ABSTRACT. Most suppliers of APL have not yet implemented the STAPL convention for transmitting workspaces from one installation to another. This report describes three workspace representations which may be used on a DECsystem-10 for this purpose. Two are partial implementations of the STAPL convention: one for level 2 of the convention, the other for level 3. The third representation is a terminal transcript file which is to be used as an input file. In addition, these representations may be used to reduce the disk storage required for APL workspaces on the DECsystem-10.

Key Words and Key Phrases: APL Workspace Interchange
APL Workspace Transfer,
Software Exchange

CR Category: 4.49

1. Introduction

This report describes three representations of an APL workspace which can be used to move workspaces from or to a *DECsystem-10*. The terminal transcript representation for moving from a *DECsystem-10* assumes that the destination machine is able to read input from a key paired *ASCII/APL* file as though the contents of the file were typed on a terminal. The second and third are implementations of levels 2 and 3 of the proposed *STAPL* convention for the interchange of workspaces [2]. Since the level 2 implementation is a key paired *ASCII/APL* representation, both the sending and receiving installations must be able to read *ASCII* files. For level 3 the workspace is converted to a bit stream and this representation is intended for exchanges between a *DECsystem-10* and a machine which does not support *ASCII* files.

Programs have been written for the *DECsystem-10* to produce the three representations and to create *APL* workspaces from the level 2 and level 3 representations of the *STAPL* convention. For the key paired *ASCII/APL* files, the programs described in [3] can be used to write the disk files on magnetic tape.

The terminal transcript and level 2 representations described here have been used to move workspaces from *DECsystem-10* to *APL/VS* on an *IBM 370/158* and to *MULTICS APL*. There is no claim, however, that the programs described in this report are completely correct and users are advised that there is no guarantee that their workspaces will be transferred correctly by any of the three methods described in this report. This caution is especially relevant to the procedures for the level 3 representation; these procedures have been tested on only a small sample of workspaces.

APLSF, as implemented on the *DECsystem-10*, has several features which are not generally available on other *APL* implementations. It is desirable to remove most of these dependencies before transmitting the workspace. Section 2 describes some of the more important changes that should be made prior to converting the workspace.

Section 3 describes the terminal transcript representation and Section 4 the level 2 and level 3 canonical representations. Section 5 gives directions for using the workspace *GENTT* to produce a terminal transcript and Section 6 directions for using the workspace *GENCRV* to generate a workspace canonical representation at level 2 or level 3. Section 7 describes the workspace *GENWS* to create a workspace from a canonical representation file on the *DECsystem-10*. Listings of these three workspaces are included as appendices.

2. Workspace Preparation

If there are locked functions in a workspace, these functions cannot be converted since it is impossible to obtain the character representations. Therefore, any locked functions which are to be transferred should be replaced by unlocked copies prior to the conversion.

2.

APLSF differs from other *APL* implementations in that two characters, carriage return and line feed, are used to separate lines; others use the single character: new line. For example, in *APLSF*:

```
'  
      C←'  
2      ρ,C  
while in APL/SV:  
'  
      C←'  
1      ρ,C
```

Since the character pair - carriage return, line feed - will not be converted into a single character by any of the three procedures discussed, it is desirable to replace all occurrences of carriage return-line feed with some other character. For the terminal transcript and level 2 representations, this must be done before a conversion is attempted; in fact, for both methods, all control characters must be replaced since the files generated are *ASCII* character files.

APLSF has a number of system functions for communicating with the file system. These include `□ASS`, `□DAS` and `□CLS`. These strings will cause errors in other implementations and should be replaced with user defined function names. More seriously in terms of transferring, the *APLSF* input/output primitives (`⊞`, `⊟`, `⊠`) are not supported in this form in other implementations. These *APLSF* characters will not be recognized as valid *APL* characters and it is, therefore, important that these *APLSF* primitives be replaced by user defined function names.

APLSF includes system commands in the domain of the execute function (`⍎`) but they are excluded in the IBM implementations. If such expressions remain in the texts of functions, they will result in an error when execution of the functions is attempted. Similarly, the *APLSF* unquote function (`⍉` or `⍊`) will cause errors and will need to be revised at some time.

The limited editing possible in *APL* is not adequate for the sorts of modifications discussed above and the function *SOS* described in [1] which permits entry into *SOS* from *APL* is particularly useful.

3. Terminal Transcript Representation

Some *APL* systems support a means of reading a file as though the lines of the file were typed on a terminal. The command `)INPUT` in *APLSF* is an example of this facility. If such a facility is available at the receiving site, it is possible to substantially reduce the computer resources required to transfer workspaces by simply writing a terminal transcript. There is, however, an important risk associated with the use of this representation. If the terminal transcript contains *APL* characters not available in the destination system, a character error will occur during the reading of the file. If this occurs, the entire transcript file must be edited it may be necessary to start the workspace restoration from the beginning. Consequently,

if this representation is to be used, the preparation discussed in Section 2 is particularly important.

In this representation, the value of a variable is formatted as a character vector and the output to the file is a sequence of assignments to the variable, followed by commands to reshape and execute, if the variable is numeric. For a function the output is a sequence of assignments to a dummy variable of the raveled canonical representation of the function definition, followed by commands to reshape and to fix. The resulting file consists of a sequence of commands which, when read into an active workspace, result in the reconstruction of the original workspace. In addition, the file contains system commands, at the beginning, to clear and to rename the workspace and, at the end, to erase the dummy variable and to save the workspace.

As an illustration, consider the workspace *MUMBLE*:

```

XX+2 5p10
XX
1    2    3    4    5
6    7    8    9   10
vY
[1]  'MANY'
[2]  'SEVERAL'
[3]  v

```

For a file line length of 30, the terminal transcript of the workspace would be:

```

)CLEAR
)NSID MUMBLE
)PP+10
)PN+80
)CT+1.13686404040747036E-13
)IO+1
)LK+''
)RL+0
)ΔRL+''
)ΔRL+ΔΔRL,'30'
)ΔRL+2ΔΔRL
)ΔΔRL'
)XX+''
)XX+XX,'1 2 3 4 5 6 7 8 9 10'
)XX+2XX
)XX+2 5pXX
)'XX'
)ΔΔET+''
)ΔΔET+ΔΔET,'Y' 'MANY''
)ΔΔET+ΔΔET,' 'SEVERAL''
)ΔΔET+3 10pΔΔET
)ΔΔET+LFX ΔΔET
)ERASE ΔΔET
)SAVE

```

4.

4. STAPL Canonical Representation

The level 2 and level 3 representations described in this report are based on the STAPL proposed convention for the interchange of workspaces. A stream of canonical representation vectors, one for each of the individuals in the workspace, is generated and written to a file. For level 2 the stream consists of key paired ASCII/APL characters and for level 3 the individuals are encoded as a stream of binary numbers. This stream is appended to a stream identifier and a translation table. The translation table is used at the destination to convert the workspace stream of bits to APL characters and for level 3 the individuals are encoded as a stream of binary numbers. This stream is appended to a stream identifier and a translation table. The translation table is used at the destination to convert the workspace stream of bits to APL characters.

The STAPL proposal defines a canonical representation vector to be of the form:

<length><type><name><space><rank><space><shape><space><elements>

In this report the types are restricted to:

C	character variable
N	numeric variable
F	function
P	pseudovisible used to describe the stream

For level 2 the stream which represents the workspace has the form:

<wsid><crv₁> . . . <crv_n> <end>

where <wsid> is the canonical representation vector naming the workspace
<crv₁> are the canonical representation vectors of the individuals
<end> is the stream termination vector

The level 2 canonical representation of the workspace MUMBLE would be:

```
16PWSID 1 6 MUMBLE9NPP 0 109N
1PW 0 808NIO 0 130NCT 0 1.13
686404040747036E 139CCLX 1 0 8
NRL 0 040FY 2 3 10 Y
'MANY' 'SEVERAL' 30NXX 2 2
5 1 2 3 4 5 6 7 8 9 108PEND 0
0
```

For level 3 the characters of the canonical representation of level 2 are further encoded as indices of a vector of the APL character set used by the sending machine. These indices are then expressed as binary numbers. For example, suppose the execute character (⍲) is the 110th element of this vector. Assuming the index origin to be zero, the execute character is represented or defined as the index 109 and would appear in the bit stream as 1101101. In this way, the APL characters of level 2 are expressed as binary numbers at level 3.

To reconstruct the *APL* characters from these binary numbers, the array *TRANSLATE* is defined. Each row of this array corresponds to an *APL* character available at the sending installation and the number of columns is the maximum number of overstrikes required to print the character (for characters requiring fewer than the maximum number of columns, the remaining columns contain the character *SPACE*). The array is then converted to a level 2 canonical representation vector and encoded as indices into the *ASCII/APL* transmission code vector. These indices are expressed in binary and this representation is attached to the beginning of the workspace bit stream. At the receiving site, this translation array is decoded using the *ASCII/APL* transmission code vector to a character string. This string is then converted to a character array which is used to construct an *APL* character vector. The *APL* character vector is then used to decode the remainder of the workspace stream to a level 2 representation.

5. Generating a Terminal Transcript

The following directions for generating a terminal transcript of a workspace assume that the workspace has been prepared as described in Section 2. All of the directions use the workspace name *MUMBLE*, all occurrences of which should be replaced by the name of the workspace to be transferred.

After the workspace is prepared, load it as an active workspace:

```
)LOAD MUMBLE
```

Next copy the terminal transcript generator and begin execution as in the following transcript:

```
)COPY APL:GENT
```

```
ΔΔINIT
```

```
DESTINATION WORKSPACE NAME: MUMBLE
```

```
OUTPUT LINE LENGTH: 80
```

The first prompt requests the name of the workspace to be used at the receiving site. If carriage return is entered, the name of the current workspace will be used (*MUMBLE*). The second prompt requests the maximum length of a file line. This will be the maximum number of *ASCII/APL* characters (the minimum should be at most 3 fewer). If carriage return is entered, the print width (*CPW*) of the active workspace will be used.

Since all output has been directed to the file, no reports are displayed on the terminal until the conversion is completed, at which time the following message is displayed:

```
FILE CREATED: MUMBLE.TSP
```

This procedure is repeated for each workspace that is to be transmitted. If desired, the separate terminal transcript files may be combined into a single file. The disk files can be written on magnetic tape using the directions given in [3].

6.

6. Generating Canonical Representations

As in Section 5 the following directions assume that the workspace has been prepared for transmission. Again the directions use the workspace name *MUMBLE* which should be replaced by the workspace name to be transmitted.

After the workspace has been prepared, load with the command:

)LOAD MUMBLE

Next copy the canonical representation generator and begin execution:

)COPY APL:GENVRV

ΔΔINIT

DESTINATION WORKSPACE NAME: MUMBLE

LEVEL OF CONVERSION:

ENTER 2(CHARACTER STREAM) OR 3(BIT STREAM):

The first prompt requests the name of the workspace to be used at the destination site. If a carriage return is entered, the name of the current workspace (*MUMBLE*) will be used. The second prompt requests whether level 2 or level 3 conversion is to be performed. If the number 3 is entered, the conversion begins. If the number 2 is entered, a third request is displayed

OUTPUT LINE LENGTH: 80

The third prompt requests the number of APL characters that are to be written on each output line. If a carriage return is entered, the print width (*□PW*) of the active workspace will be used. In deciding on the line length, allow for a substantial increase in the actual line length due to overstruck characters.

7. Restoring Canonical Representations

The following directions assume that the canonical representation file to be converted is of a single workspace. Load the workspace generator and begin execution:

)LOAD GENES

ΔΔINIT

FILE TO BE CONVERTED: MUMBLE

TYPE OF FILE:

ENTER 2(CHARACTER STREAM) OR 3(BIT STREAM):

The first prompt requests the name and extension (if any) of the file. The second prompt requests whether the file is an *ASCII/APL* character stream or a bit stream. If the number 2 is entered, a third request is displayed:

FILE LINE LENGTH:

The line length entered should be that used when the file was generated.

After this initial dialogue, the reconstruction of the workspace begins. For a bit stream, the initial phase is a check that the stream identifier is correct. If it is not, a message is displayed that the file does not conform to *STAPL* format conventions and the conversion is terminated. If it is correct, the translation table is reconstructed using the *ASCII/APL*

transmission vector and then written on a temporary file *MUMOnn.CRV* (where 'nn' is a two digit number). This file is then read to create the APL character vector. If there are APL characters which are not recognized by *APLSF*, a character error will occur at this point. Should this occur, the temporary file can be modified and execution resumed at $\Delta\Delta TCVT[3]$.

Following this phase of the bit stream reconstruction, the workspace is converted to a character stream and written on the file *MUMOnn.CRV*.

The character stream is read in blocks of 10 lines and the individual canonical representation vectors are extracted. As the APL individuals are reconstructed from the canonical representation vectors, the names are displayed on the terminal.

When the individuals have been reconstructed, the conversion is terminated with the following messages:

*THE ABOVE INDIVIDUALS HAVE BEEN RECREATED
TO COMPLETE THE WORKSPACE RECONSTRUCTION;
ENTER THE FOLLOWING:*

*)WSID MUMBLE
)ERASE $\Delta\Delta GENWS$
)SAVE*

where $\Delta\Delta GENWS$ is the group of global variables and functions of *GENWS*.

8.

APPENDIX A

WORKSPACE GENNT
TERMINAL TRANSCRIPT GENERATOR

System Variables:

$\square IO \leftrightarrow 1$

Variables:

$\Delta\Delta QS$ is used in determining the number of APL characters to be written on an output line.

47 $\rho\Delta\Delta QS$
 $\Delta\Delta QS$
a!b\A Ie v x \ / e p q : v b l e g $\Delta\Delta ABCDEFGHIJKLMNOPQRSTUVWXYZ$

GENTT

ΔΔINIT

```

V ΔΔINIT;ΔΔWS;ΔΔRL;ΔΔIO
[1] ΔΔIO←ΔΔIO
[2] ΔΔIO←1
[3] 'DESTINATION WORKSPACE NAME: A'
[4] ΔΔWS←Δ
[5] ΔΔWS←2 5 -4[1+1≠ρΔΔWS,' ']+')WSID ΔΔWS'
[6] 'TERMINAL TRANSCRIPT LINE LENGTH: A'
[7] ΔΔRL←Δ
[8] ΔΔRL←2 3 -5[1+1≠ρΔΔRL,' ']+')PW 2ΔΔRL'
[9] ΔΔGENW

```

ΔΔGENW

```

V ΔΔGENW;ΔΔX;ΔΔE
[1] A ΔΔGENW CONVERTS THE WORKSPACE ΔΔWS TO A TERMINAL
[2] A TRANSCRIPT ON THE KEY PAIRED ASCII FILE ΔΔWS.TSP
[3] 2')OUTPUT ',ΔΔWS,'.TSP'
[4] ' )CLEAR'
[5] ' )WSID ',ΔΔWS
[6] ' )PP+',ΔΔPP
[7] ' )PW+',ΔΔPW
[8] ΔΔPP←18
[9] ΔΔPW←ΔΔRL
[10] ' )CT+',ΔΔCT
[11] ' )IO+',ΔΔΔIO
[12] ' )LX+',ΔΔLX,' '
[13] ' )RL+',ΔΔRL
[14] A CONVERT VARIABLES IN WORKSPACE
[15] ΔΔX←ΔNL 2
[16] ΔΔX←(Δ/ΔΔXV.≠((1+ρΔΔX),5)+Q 5 5 ρ'ΔΔWS ΔΔX ΔΔCE ΔΔQS ΔΔIO ')+ΔΔX
[17] ΔΔGENW1:→(0=ρ,ΔΔX)/ΔΔGENW2
[18] ΔΔTRV ΔΔTRIM ΔΔX[1;]
[19] ΔΔX← 1 0 +ΔΔX
[20] →ΔΔGENW1
[21] A CONVERT FUNCTIONS IN WORKSPACE
[22] ΔΔGENW2:ΔΔX←ΔNL 3
[23] ΔΔE←'ΔΔGENW ΔΔTRIM ΔΔSTRP 'ΔΔTRV ΔΔTRF ΔΔDPLQ ΔΔQDCR ΔΔINIT'
[24] ΔΔX←(Δ/ΔΔXV.≠((1+ρΔΔX),8)+Q 8 7 ρΔΔE)+ΔΔX
[25] ΔΔGENW3:→(0=ρ,ΔΔX)/ΔΔGENW4
[26] ΔΔTRF ΔΔTRIM ΔΔX[1;]
[27] ΔΔX← 1 0 +ΔΔX
[28] →ΔΔGENW3
[29] ΔΔGENW4:' )ERASE ΔΔET'
[30] ' )SAVE'
[31] 2')OUTPUT'
[32] 'FILE CREATED: ',ΔΔWS,'.TSP'

```

10.

GENPT

ΔΔTTRV

```

V ΔΔTTRV ΔΔW;ΔΔC;ΔΔS;ΔΔL
[1] A ΔΔTTRV PRODUCES A TERMINAL TRANSCRIPT OF A
[2] A VARIABLE ASSIGNMENT
[3] ΔΔC←ΔΔW
[4] ΔΔS←▽ρΔΔC
[5] ΔΔC←ΔΔDPLQ▽,ΔΔC
[6] A ASSIGN TO VARIABLE NAME ITS VALUE REPRESENTED
[7] A AS A CHARACTER VECTOR
[8] ΔΔW,'+''''
[9] ΔΔTTRV1:→(0=ρΔΔC)/ΔΔTTRV2
[10] A DETERMINE LENGTH OF CHARACTER STRING TO FILL FILE LINE
[11] ΔΔW←1+((+\\1+(ΔΔC∈ΔΔQS)×2)>(ρΔΔC)[PW-4+2×ρ,ΔΔW])1
[12] A CHECK CHARACTER STRING DOES NOT
[13] A CONTAIN ODD NUMBER OF QUOTES
[14] ΔΔW+ΔΔW-0≠2|+/(ΔΔW+ΔΔC)='''
[15] ΔΔW,'+',ΔΔW,',','',(ΔΔW+ΔΔC),''''
[16] ΔΔC+ΔΔW+ΔΔC
[17] →ΔΔTTRV1
[18] A IF NUMERIC, CONVERT VALUE
[19] ΔΔTTRV2:→(' '=1↑0ρ,ΔΔW)/ΔΔTTRV3
[20] ΔΔW,'+Δ',▽Δ 4 1[1+0=ρ,ΔΔW]↑'ΔΔW 0'
[21] A RESHAPE VARIABLE
[22] ΔΔTTRV3:Δ(0=ρΔΔS)/'ΔΔW','+',ΔΔS,'0','ΔΔW'
[23] ''',ΔΔW,'''

```

ΔΔTTRF

```

V ΔΔTTRF ΔΔA;ΔΔW;ΔΔS
[1] A ΔΔTTRF PRODUCES A TERMINAL TRANSCRIPT OF THE FUNCTION ΔΔA
[2] ΔΔA+ΔΔQDCR ΔΔA
[3] ΔΔS←ρΔΔA
[4] ΔΔA+ΔΔDPLQ,ΔΔA
[5] 'ΔΔET←''''
[6] ΔΔTTRF1:→(0=ρΔΔA)/ΔΔTTRF2
[7] A DETERMINE LENGTH OF CHARACTER STRING TO FILL FILE LINE
[8] ΔΔW←1+((+\\1+(ΔΔA∈ΔΔQS)×2)>(ρΔΔA)[PW-12])1
[9] A CHECK THAT CHARACTER STRING DOES NOT
[10] A CONTAIN ODD NUMBER OF QUOTES
[11] ΔΔW+ΔΔW-0≠2|+/(ΔΔW+ΔΔA)='''
[12] 'ΔΔET+ΔΔET,','',(ΔΔW+ΔΔA),''''
[13] ΔΔA+ΔΔW+ΔΔA
[14] →ΔΔTTRF1
[15] A RESHAPE FUNCTION DEFINITION AS A CANONICAL REPRESENTATION
[16] ΔΔTTRF2:'ΔΔET+',(▽ΔΔS),'0ΔΔET'
[17] A FIX THE FUNCTION
[18] 'ΔΔET←ΔFX ΔΔET'

```


GENCRV

ΔΔINIT

```

V ΔΔINIT;ΔΔYS;ΔΔRL;ΔΔCH;ΔΔLC;ΔΔES;ΔΔIQ
[1] ΔΔIQ←ΔIO
[2] ΔIO←0
[3] 'DESTINATION WORKSPACE NAME: A'
[4] ΔΔYS←Δ
[5] Δ(1=0ΔΔYS,' ')/'ΔΔYS+Δ''WSID''
[6] 'LEVEL OF CONVERSION:'
[7] ' ENTER 2(CHARACTER STREAM) OR 3(BIT STREAM): A'
[8] ΔΔLC←-2+Δ
[9] →ΔΔLC/ΔΔINIT3
[10] A INITIALIZATION FOR CHARACTER STREAM FILE
[11] 'FILE LINE LENGTH: A'
[12] ΔΔRL←Δ
[13] ΔΔRL←Δ 3 -5[1=0ΔΔRL,' ']+'BPW ΔΔRL'
[14] ΔΔCH←DASS ΔΔYS,'.CRV/AS'
[15] ΔΔBY←''
[16] →ΔΔINIT1
[17] A INITIALIZATION FOR BIT STREAM FILE
[18] ΔΔINIT3:ΔΔCH←DASS ΔΔYS,'.BRV/BU'
[19] A DETERMINE FRAMESIZE
[20] ΔΔES←(2*0ΔΔAL/
[21] ΔΔRL←BPW
[22] ΔΔINIT1:ΔΔCCR/
[23] DASS ΔΔCH
[24] 'CONVERSION COMPLETED:'
[25] ' FILE CREATED: ',ΔΔYS, 7 -7[ΔΔLC]+'.CRV/AS .BRV/BU'

```

12.

APPENDIX B

WORKSPACE GENCRV

CANONICAL REPRESENTATION GENERATOR

System Variables:

$\square IO \leftrightarrow 0$

Variables:

$\Delta\Delta V$ is the APL character vector

```
      p $\Delta\Delta V$ 
142
       $\Delta\Delta V$ 
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789,./[ ]{ }~<=>≠∇∧-÷
$→*o1†~pεω?αΓ∇Δ°'□()}\:;|T⊥U∩▷◁!$%&⊖⊗∧∨/⊥⊞⊠⊡⊢⊣⊤⊥⊦⊧⊨⊩⊪⊫⊬⊭⊮⊯⊰⊱⊲⊳⊴⊵⊶⊷⊸⊹⊺⊻⊼⊽⊾⊿⊿
 $\Delta\Delta\Delta V$ 
```

$\Delta\Delta BV$ is the level 3 representation of the stream identifier and the translation table. As described in the text, the translation table is an array, the rows corresponding to the APL characters in $\Delta\Delta V$. The number of columns is the maximum number of overstrikes required to print the APL characters. Thus

row 2 would be SPACE, 'A'

and row 142 would be '_', 'Z'

$\Delta\Delta BV$ is formed by first constructing the level 2 canonical representation vector

$CRV \leftarrow ('O' \Delta\Delta PREP 'WSIS'), TT \Delta\Delta PREP 'TRANSLATE'$

where TT is the translation table array

$\Delta\Delta BV$ is then defined as

$,O(8p2)T\Delta\Delta ASCII\Delta APL_1CRV$

where $\Delta\Delta ASCII\Delta APL$ is the ASCII/APL

transmission code vector

($\Delta\Delta ASCII\Delta APL$ is defined in the workspace *GENWS*)

In generating the bit stream, the incoming character stream to the function $\Delta\Delta OUTF$ is processed in blocks of 512 characters. The size of this block may need to be modified.

GENCRV

ΔΔCCRZ

```

V  ΔΔCCRZ;ΔΔX;ΔΔY
[1]  A  ΔΔCCRZ GENERATES A WORKSPACE CANONICAL REPRESENTATION VECTOR
[2]  A  CONFORMING TO THE STAPL PROPOSED CONVENTION FOR LEVEL 2
[3]  ΔΔIO←ΔΔIO
[4]  ΔΔX←(ΔΔYS ΔΔPREP 'WSID'),(ΔΔVREP 'LPP'),(ΔΔZREP 'EPW'),ΔΔVREP 'LIO'
[5]  ΔΔPP←18
[6]  ΔΔPN←ΔΔARL
[7]  ΔΔIO←0
[8]  ΔΔQUTE ΔΔX,(ΔΔZREP 'LCT'),(ΔΔVREP 'ELX'),ΔΔVREP 'LRL'
[9]  A  GENERATION OF FUNCTION CANONICAL REPRESENTATION VECTORS
[10] ΔΔX←ΔΔNL 3
[11] ΔΔY←'ΔΔINIT ΔΔQUTE ΔΔTRIM ΔΔSTER ΔΔFEREP ΔΔVREP ΔΔPREP ΔΔCCRZ ΔΔQDCR '
[12] ΔΔX←(ΔΔX√.≠((1+ρΔΔX),9)+Φ 9 7 ρΔΔY)ΔΔX
[13] ΔΔCCRZ1:→(0=ρ,ΔΔX)/ΔΔCCRZ2
[14] ΔΔQUTE ΔΔFEREP,ΔΔTRIM ΔΔX[0;]
[15] 'FUNCTION CONVERTED: ',ΔΔX[0;]
[16] ΔΔY←LEX ΔΔX[0;]
[17] ΔΔX← 1 0 +ΔΔX
[18] →ΔΔCCRZ1
[19] A  GENERATION OF VARIABLE CANONICAL REPRESENTATION VECTORS
[20] ΔΔCCRZ2:ΔΔX←ΔΔNL 2
[21] ΔΔY←'ΔΔX ΔΔX ΔΔARL ΔΔCH ΔΔAV ΔΔBY ΔΔES ΔΔYS ΔΔLC ΔΔIO '
[22] ΔΔX←(ΔΔX√.≠((1+ρΔΔX),10)+Φ 10 5 ρΔΔY)ΔΔX
[23] ΔΔCCRZ3:→(0=ρ,ΔΔX)/ΔΔCCRZ4
[24] ΔΔQUTE ΔΔVREP,ΔΔTRIM ΔΔX[0;]
[25] 'VARIABLE CONVERTED: ',ΔΔX[0;]
[26] ΔΔY←LEX ΔΔX[0;]
[27] ΔΔX← 1 0 +ΔΔX
[28] →ΔΔCCRZ3
[29] ΔΔCCRZ4:ΔΔQUTE '0' ΔΔPREP 'END'

```


GENCR/

ΔΔQUTE

```

V ΔΔQUTE ΔΔV;ΔΔP;ΔΔR;ΔΔS;ΔΔT;ΔΔB
[1] A IF CONVERSION COMPLETED, FLAG TO OUTPUT REMAINDER OF STREAM
[2] ΔΔS←Δ/'SPEND'≠5+ΔΔV
[3] A ADD CURRENT VECTOR TO ACCUMULATED STREAM BUFFER
[4] A (IF BIT STREAM, ITERATIVELY ADD BLOCKS OF 512
[5] A CHARACTERS CONVERTED TO BIT VECTORS)
[6] ΔΔB←512
[7] ΔΔQUTE1:ΔΔBV←ΔΔBV,2 3 -31[ΔΔLC]←'ΔΔV ,Q(ΔΔFS02)τΔΔAV,(ΔΔB[0ΔΔV]+ΔΔV'
[8] ΔΔV←2 2 -8[ΔΔLC]←'''' ΔΔB+ΔΔV'
[9] A OUTPUT FULL LINES (CHARACTER STREAM)
[10] A OR FULL WORDS (BIT STREAM) OF ACCUMULATED STREAM
[11] ΔΔR←2 4 -2[ΔΔLC]←'ΔΔRL 36'
[12] ΔΔT←(-ΔΔR[0ΔΔBV]+ΔΔBV
[13] →(ΔΔR>0ΔΔBV)/ΔΔQUTE2
[14] ΔΔBV←(-ΔΔR[0ΔΔBV]+ΔΔBV
[15] ΔΔBV←2 36 -13[ΔΔLC]←'(((0ΔΔBV)÷ΔΔRL),ΔΔRL)0ΔΔBV)⊕[5]ΔΔCH ΔΔBV⊕ΔΔCH,0,2'
[16] ΔΔQUTE2:ΔΔBV←ΔΔT
[17] →(0≠p,ΔΔV)/ΔΔQUTE1
[18] →ΔΔS/0
[19] A IF CONVERSION COMPLETED, OUTPUT REMAINDER OF STREAM
[20] ΔΔB←2 12 -13[ΔΔLC]←'ΔΔBV⊕[5]ΔΔCH ΔΔBV⊕ΔΔCH,0,2'

```

V

16.

GENCKV

ΔΔZREP

```

V  ΔΔR←ΔΔZREP ΔΔA
[1] A  ΔΔA IS THE NAME OF THE VARIABLE
[2] A  GET VALUE
[3]   ΔΔR←ΔΔA
[4] A  CATENATE NAME, RANK, SHAPE VECTOR, AND VALUE
[5]   ΔΔR←ΔΔA,'',(▽(ρρΔΔR),ρΔΔR),'',▽,ΔΔR
[6] A  APPEND DATA TYPE
[7]   ΔΔR←'NC'[(11)+' '=1+0ρΔΔA],ΔΔR
[8] A  APPEND VECTOR LENGTH
[9]   ΔΔR←(▽ρΔΔR),ΔΔR

```

V

ΔΔFEREP

```

V  ΔΔR←ΔΔFEREP ΔΔA
[1] A  ΔΔA IS THE NAME OF THE FUNCTION TO BE CONVERTED
[2] A  GET CANONICAL REPRESENTATION
[3]   ΔΔR←ΔΔQDCR ΔΔA
[4] A  APPEND DATA TYPE, RANK, AND SHAPE VECTOR
[5]   ΔΔR←'F',ΔΔA,'',(▽(ρρΔΔR),ρΔΔR),'',ΔΔR
[6] A  APPEND VECTOR LENGTH
[7]   ΔΔR←(▽ρΔΔR),ΔΔR

```

V

ΔΔPREP

```

V  ΔΔR←ΔΔV ΔΔPREP ΔΔA
[1] A  GENERATE PSEUDO-VARIABLE
[2] A  ΔΔA IS THE NAME OF THE PSEUDO-VARIABLE AND
[3] A  ΔΔV IS THE VALUE
[4] A  CATENATE NAME, RANK, SHAPE VECTOR, AND VALUE
[5]   ΔΔR←'P',ΔΔA,'',(▽(ρρΔΔV),ρΔΔV),'',▽,ΔΔV
[6] A  APPEND VECTOR LENGTH
[7]   ΔΔR←(▽ρΔΔR),ΔΔR

```

V

APPENDIX C

WORKSPACE GENWS

WORKSPACE RECONSTRUCTION FROM
CANONICAL REPRESENTATION

System Variables:

$\square IO \leftrightarrow 0$

Variables:

$\Delta\Delta ASCII\Delta APL$ is a partial *ASCII/APL* transmission code vector containing the printable *ASCII/APL* characters and the *ASCII/APL* character *NUL*. If the incoming translation table contains control characters, $\Delta\Delta ASCII\Delta APL$ should be extended, using the standard decimal *ASCII* representation to determine the indices.

The function $\Delta\Delta CRCVT$ extracts individual canonical representation vectors from the partial character stream in core. If the stream is too short to extract the next vector, $\Delta\Delta CFIN$ reads blocks of 5 lines of the *ASCII* file until the stream is sufficiently long. The number of lines read by $\Delta\Delta CFIN$ may need to be modified.

GENWS

ΔΔINIT

```

V ΔΔINIT;ΔΔFN;ΔΔRL;ΔΔCH;ΔΔWS;ΔΔIO
[1] 'FILE TO BE CONVERTED: A'
[2] ΔΔFN+□
[3] 'TYPE OF FILE:'
[4] '    ENTER 2(CHARACTER STREAM) OR 3(BIT STREAM): A'
[5] +('3'=□)/ΔΔINIT3
[6] 'FILE LINE LENGTH: A'
[7] ΔΔRL+□
[8] ΔΔCH+□ASS ΔΔFN,'/AS'
[9] +ΔΔINIT1
[10] ΔΔINIT3:ΔΔBFIN
[11] +ΔΔTRANS/O
[12] ΔΔCH+ΔΔBCVT
[13] ΔΔINIT1:ΔΔCRCVT ''
[14] ΔΔIO+ΔΔIO
[15] ''
[16] 'THE ABOVE INDIVIDUALS HAVE BEEN RECREATED.'
[17] 'TO COMPLETE THE WORKSPACE RECONSTRUCTION.'
[18] 'ENTER THE FOLLOWING:'
[19] '    )WSID ',ΔΔWS
[20] '    )ERASE ΔΔGENWS'
[21] '    )SAVE'
V

```


GENWS

ΔΔTRANS

```

V ΔΔERR←ΔΔTRANS;ΔΔBS;ΔΔCV;ΔΔTL
[1] A ΔΔTRANS CHECKS THE STREAM IDENTIFIER
[2] A AND EXTRACTS THE TRANSLATION TABLE
[3] ΔΔCV←ΔΔASCIIΔAPL[(802)10 15 8 0120+ΔΔBF]
[4] →('9PWSIS 0 0'=10+ΔΔCV)/ΔΔTRANS1
[5] ' FILE DOES NOT CONFORM TO STAPL FORMAT CONVENTIONS '
[6] ' TO INSPECT THE FILE, '
[7] ΔΔFN, ' HAS BEEN ASSIGNED TO THE VARIABLE ΔΔBF '
[8] ' AND THE ATTEMPTED CONVERSION OF THE FIRST 15 CHARACTERS '
[9] ' HAS BEEN ASSIGNED TO ΔΔCV '
[10] ΔΔERR←1
[11] →0
[12] A CONVERT THE TRANSLATION TABLE FROM BINARY
[13] A TO A CHARACTER VECTOR
[14] ΔΔTRANS1:ΔΔL←ΔΔCV[9+11+(-5+ΔΔCV),1'P']
[15] ΔΔL←8×(10+ΔΔL+0ΔΔL)
[16] ΔΔCV←ΔΔASCIIΔAPL[(802)10((-10+ΔΔL÷8),8)080+ΔΔL+ΔΔBF]
[17] ΔΔBF←ΔΔL+ΔΔBF
[18] A CONSTRUCT THE TRANSLATION TABLE ARRAY
[19] ΔΔTCVT ΔΔCV
[20] A FOR EACH ROW OF THE TRANSLATION TABLE,
[21] A INSERT BACKSPACE BETWEEN EACH CHARACTER
[22] ΔΔTT←,ΔΔTRANSLATE[;ΔΔC+0]
[23] ΔΔBS←(0ΔΔTRANSLATE)[0]0ΔΔAV[98]
[24] ΔΔTRANS2:→((0ΔΔTRANSLATE)[1]=ΔΔC+ΔΔC+1)/ΔΔTRANS3
[25] ΔΔTT+ΔΔTT,ΔΔBS,,ΔΔTRANSLATE[;ΔΔC]
[26] →ΔΔTRANS2
[27] ΔΔTRANS3:ΔΔTCVTQ((-1+2×(0ΔΔTRANSLATE)[1]),(0ΔΔTRANSLATE)[0])0ΔΔTT
[28] ΔΔERR←0

```

ΔΔTCVT

```

V ΔΔTCVT ΔΔA;ΔΔCH;ΔΔI
[1] A ΔΔTCVT CONVERTS THE TRANSLATION TABLE
[2] A TO THE APL CHARACTER VECTOR
[3] ΔΔCH←ΔΔASS((4[0ΔΔFN]+ΔΔFN),(-2+0ΔΔUL),'.CRV/AS')
[4] ΔΔA←ΔΔAΔ[5]ΔΔCH
[5] ΔΔLS ΔΔCH
[6] ΔΔI←0
[7] ΔΔTRANSLATE←''
[8] ΔΔTCVT1:ΔΔTRANSLATE←ΔΔTRANSLATE,1+Δ[5]ΔΔCH
[9] →((0ΔΔA)[0]>ΔΔI+ΔΔI+1)/ΔΔTCVT1
[10] ΔΔAS ΔΔCH

```

GENVS

 $\Delta\Delta\text{CRCVT}$

```

V   $\Delta\Delta\text{CRCVT}$   $\Delta\Delta\text{W};\Delta\Delta\text{P};\Delta\Delta\text{L};\Delta\Delta\text{V};\Delta\Delta\text{T};\Delta\Delta\text{N};\Delta\Delta\text{R};\Delta\Delta\text{S}$ 
[1]  A   $\Delta\Delta\text{CRCVT}$  CREATES A WORKSPACE FROM THE
[2]  A  CHARACTER STREAM OF THE WORKSPACE
[3]   $\Delta\Delta\text{CRCVT1}:\mathbf{2}(10>\rho\Delta\Delta\text{W})/'\Delta\Delta\text{W}+\Delta\Delta\text{CFIN } \Delta\Delta\text{W}'$ 
[4]  A  DETERMINE LENGTH OF AN INDIVIDUAL
[5]  A  CANONICAL REPRESENTATION VECTOR
[6]   $\Delta\Delta\text{L}+(\text{L}/(10+\Delta\Delta\text{W}))\mathbf{1}'\text{NCFP}'+\Delta\Delta\text{V}$ 
[7]   $\Delta\Delta\text{CRCVT2}:\mathbf{+}((\rho\Delta\Delta\text{W})\geq\mathbf{2}\Delta\Delta\text{L})/\Delta\Delta\text{CRCVT3}$ 
[8]   $\Delta\Delta\text{V}+\Delta\Delta\text{CFIN } \Delta\Delta\text{W}$ 
[9]   $\rightarrow\Delta\Delta\text{CRCVT2}$ 
[10] A  EXTRACT INDIVIDUAL CANONICAL REPRESENTATION VECTOR
[11]  $\Delta\Delta\text{CRCVT3}:\Delta\Delta\text{W}+(\rho\Delta\Delta\text{V}+\Delta\Delta\text{W}[(\rho\Delta\Delta\text{L})+\mathbf{1}\mathbf{2}\Delta\Delta\text{L}])+\rho\Delta\Delta\text{L}+\Delta\Delta\text{V}$ 
[12]  $\Delta\Delta\text{P}+(\Delta\Delta\text{V}=\mathbf{' '})/\mathbf{1}\rho\Delta\Delta\text{V}$ 
[13] A  GET TYPE OF INDIVIDUAL
[14]  $\Delta\Delta\text{T}+\mathbf{1}+\Delta\Delta\text{V}$ 
[15] A  GET NAME
[16]  $\Delta\Delta\text{N}+\mathbf{1}+\Delta\Delta\text{P}[\mathbf{0}]+\Delta\Delta\text{V}$ 
[17] A  GET RANK
[18]  $\Delta\Delta\text{R}+\mathbf{2}(1+\rho\Delta\Delta\text{N})+\Delta\Delta\text{P}[\mathbf{1}]+\Delta\Delta\text{V}$ 
[19] A  GET SHAPE VECTOR
[20]  $\mathbf{2}'\Delta\Delta\text{S}+\mathbf{'}, \mathbf{2}^{-22}[\Delta\Delta\text{R}=\mathbf{0}]+\mathbf{'10 } \mathbf{2}\Delta\Delta\text{P}[\mathbf{1}]+\Delta\Delta\text{P}[\mathbf{1}+\Delta\Delta\text{R}]+\Delta\Delta\text{V}'$ 
[21]  $\Delta\Delta\text{V}+(\mathbf{1}+\Delta\Delta\text{P}[\mathbf{1}+\Delta\Delta\text{R}])+\Delta\Delta\text{V}$ 
[22]  $\rightarrow(\Delta\Delta\text{T}=\mathbf{'P'})/\Delta\Delta\text{CRCVT5}$ 
[23] A  IF NUMERIC VARIABLE, CONVERT VALUE TO NUMBER
[24]  $\mathbf{2}(\Delta\Delta\text{T}=\mathbf{'N'})/'\Delta\Delta\text{V}+\mathbf{24}^{-1}[\mathbf{0}=\rho\Delta\Delta\text{V}]+\mathbf{'2}\Delta\Delta\text{V } \mathbf{0}''$ 
[25] A  SAVE BIO OF WORKSPACE
[26]  $\rightarrow(\mathbf{'BIO'}+\mathbf{3}+\Delta\Delta\text{N})/\Delta\Delta\text{CRCVT4}$ 
[27]  $\Delta\Delta\text{IO}+\Delta\Delta\text{V}$ 
[28]  $\Delta\Delta\text{N}$ 
[29]  $\rightarrow\Delta\Delta\text{CRCVT1}$ 
[30] A  ASSIGN VALUE TO VARIABLE OR FIX FUNCTION DEFINITION
[31]  $\Delta\Delta\text{CRCVT4}:\mathbf{2} \mathbf{15}^{-11}[\Delta\Delta\text{T}=\mathbf{'F'}]+\mathbf{'2}\Delta\Delta\text{N},\mathbf{' }+\Delta\Delta\text{S}\rho\Delta\Delta\text{V}''\mathbf{EX } \Delta\Delta\text{S}\rho\Delta\Delta\text{V}'$ 
[32]  $\mathbf{2}(\Delta\Delta\text{T}=\mathbf{'F'})/'\Delta\Delta\text{N}'$ 
[33]  $\rightarrow\Delta\Delta\text{CRCVT1}$ 
[34] A  PROCESS PSEUDOVARIABLES.
[35]  $\Delta\Delta\text{CRCVT5}:\mathbf{2}(\wedge/'\text{WSID'}=\mathbf{4}+\Delta\Delta\text{N})/'\Delta\Delta\text{WS}+\Delta\Delta\text{V}'$ 
[36] A  IF BIT STREAM, CONSTRUCT TRANSLATE TABLE
[37]  $\rightarrow(\mathbf{'TRANSLATE'}+\mathbf{9}+\Delta\Delta\text{N})/\Delta\Delta\text{CRCVT6}$ 
[38]  $\Delta\Delta\text{TRANSLATE}+(\mathbf{2}\Delta\Delta\text{S})\rho\Delta\Delta\text{V}$ 
[39]  $\rightarrow\mathbf{0}$ 
[40]  $\Delta\Delta\text{CRCVT6}:\rightarrow(\mathbf{'END'}+\mathbf{3}+\Delta\Delta\text{N})/\Delta\Delta\text{CRCVT1}$ 

```

GENVS

ΔΔBFIN

```

V ΔΔBFIN;ΔΔCH
[1] A ΔΔBFIN READS THE BIT STREAM REPRESENTATION
[2] A OF THE WORKSPACE
[3] ΔΔCH←[ASS ΔΔFN,'/BU'
[4] ΔΔBF←[ΔΔCH,0,2,(LFLS ΔΔCH)[2]×36
[5] [DAS ΔΔCH

```

ΔΔBCVT

```

V ΔΔCH←ΔΔBCVT;ΔΔBL;ΔΔT;[PV;ΔΔFS
[1] A ΔΔBCVT CONVERTS THE BINARY STREAM TO A CHARACTER
[2] A STREAM USING THE APL CHARACTER VECTOR
[3] ΔΔCH←[ASS((4[pΔΔFN)+ΔΔFN),(-2+V[UL),'CRV/AS'
[4] ΔΔRL←[PV+128
[5] ΔΔFS←[2[pΔΔTRANSLATE
[6] ΔΔBL←ΔΔFS×512
[7] ΔΔBCVT1:ΔΔT←ΔΔTRANSLATE[(ΔΔFS÷2)1Q(512,ΔΔFS)pΔΔBL+ΔΔBF]
[8] ΔΔT←(4 128 pΔΔT)B[5]ΔΔCH
[9] ΔΔBF←ΔΔBL+ΔΔBF
[10] →(0≠pΔΔBF)/ΔΔBCVT1
[11] [CLS ΔΔCH

```

ΔΔCFIN

```

V ΔΔR←ΔΔCFIN ΔΔC;ΔΔL
[1] A ΔΔCFIN READS A BLOCK OF 5 FILE LINES OF
[2] A CHARACTER STREAM REPRESENTATION OF THE WORKSPACE
[3] ΔΔR←ΔΔC
[4] ΔΔL←0
[5] ΔΔCFIN1:ΔΔV←B[5]ΔΔCH
[6] →(0 75 =o,ΔΔV)/0
[7] ΔΔR←ΔΔR,ΔΔRL+ΔΔV
[8] →(5>ΔΔL+ΔΔL+1)/ΔΔCFIN1

```

22.

APPENDIX D

UTILITY FUNCTIONS

The following functions are used in the workspaces. $\Delta\Delta QDCR$ is included because, in the version of *APLSF* on which these programs were run, the system function $\square CR$ did not perform correctly.

The four utility functions assume $\square IO$ to be 1 and minor modifications are required if $\square IO$ is 0.

$\Delta\Delta QDCR$

```

V   $\Delta\Delta R + \Delta\Delta QDCR \Delta\Delta A; \Delta\Delta D; \Delta\Delta L; \Delta\Delta I$ 
[1]  A   $\Delta\Delta QDCR$  IS EQUIVALENT TO
[2]  A  THE SYSTEM FUNCTION  $\Delta\Delta R$ 
[3]     $\Delta\Delta D + 1 + \Delta\Delta A$ 
[4]     $\Delta\Delta I + \Delta\Delta D = \Delta\Delta V[102]$ 
[5]     $\Delta\Delta L + 10$ 
[6]     $\Delta\Delta QDCR1: \rightarrow \Delta\Delta QDCR2 \times 13 \geq 0 \Delta\Delta I$ 
[7]     $\Delta\Delta L + \Delta\Delta L, (\Delta\Delta I, 1) - 1$ 
[8]     $\Delta\Delta I + ((1 + \Delta\Delta L) + 2) + \Delta\Delta I$ 
[9]     $\rightarrow \Delta\Delta QDCR1$ 
[10]    $\Delta\Delta QDCR2: \Delta\Delta R + ((0 \Delta\Delta L), \lceil / \Delta\Delta L \rceil 0) ' '$ 
[11]    $\Delta\Delta I + 1$ 
[12]    $\Delta\Delta QDCR3: \rightarrow (\Delta\Delta I > 0 \Delta\Delta L) / 0$ 
[13]    $\Delta\Delta R[\Delta\Delta I; 1 \Delta\Delta L[\Delta\Delta I]] + \Delta\Delta L[\Delta\Delta I] + \Delta\Delta STRP \Delta\Delta L[\Delta\Delta I] + \Delta\Delta D$ 
[14]    $\Delta\Delta D + (\Delta\Delta L[\Delta\Delta I] + 2) + \Delta\Delta D$ 
[15]    $\Delta\Delta I + \Delta\Delta I + 1$ 
[16]    $\rightarrow \Delta\Delta QDCR3$ 

```

 $\Delta\Delta STRP$

```

V   $\Delta\Delta R + \Delta\Delta STRP \Delta\Delta A$ 
[1]  A   $\Delta\Delta STRP$  REMOVES LEADING BLANKS
[2]  A  FROM THE CHARACTER VECTOR  $\Delta\Delta A$ 
[3]     $\Delta\Delta R +, \Delta\Delta A$ 
[4]     $\Delta\Delta R + (((\Delta\Delta R \in ' ')) 10) - 1) + \Delta\Delta R$ 

```

 $\Delta\Delta TRIM$

```

V   $\Delta\Delta R + \Delta\Delta TRIM \Delta\Delta A$ 
[1]  A   $\Delta\Delta TRIM$  REMOVES TRAILING BLANKS
[2]  A  FROM THE CHARACTER VECTOR  $\Delta\Delta A$ 
[3]     $\Delta\Delta R + \Phi \Delta\Delta STRP \Phi \Delta\Delta A$ 

```

 $\Delta\Delta DPLQ$

```

V   $\Delta\Delta R + \Delta\Delta DPLQ \Delta\Delta A; \Delta\Delta I$ 
[1]  A   $\Delta\Delta DPLQ$  INSERTS A SECOND QUOTE AFTER EACH
[2]  A  OCCURRENCE OF A QUOTE IN STRING  $\Delta\Delta A$ 
[3]     $\Delta\Delta R + \Delta\Delta A$ 
[4]     $\Delta\Delta I + 1$ 
[5]     $\Delta\Delta DPLQ1: \rightarrow ((0 \Delta\Delta R) < \Delta\Delta I + \Delta\Delta I + 1 + ((\Delta\Delta I + 1) + \Delta\Delta R) 1 ' ' ' ' ) / 0$ 
[6]     $\Delta\Delta R + (\Delta\Delta I + \Delta\Delta R), ' ' ' ', \Delta\Delta I + \Delta\Delta R$ 
[7]     $\rightarrow \Delta\Delta DPLQ1$ 

```

REFERENCES

- [1] R. J. Orgass, Transferring Files to an IBM Machine, Technical Memorandum No. APLAD14, July 28, 1978, University of Arizona.
- [2] D. Cartwright, Proposed Standard for the Interchange of APL Workspaces, Dec 1977, *APL Quote Quad*.
- [3] R. J. Orgass and G. M. Uhler, Using *SOS* in *APLSF*, Technical Memorandum, September 14, 1977, University of Arizona.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR. 79-0238	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) TRANSFER OF APL WORKSPACES: A USEFUL SOLUTION		5. TYPE OF REPORT & PERIOD COVERED Interim
7. AUTHOR(s) Karl Rautenkranz and Richard J. Orgass		6. PERFORMING ORG. REPORT NUMBER CS78006-T
9. PERFORMING ORGANIZATION NAME AND ADDRESS Virginia Polytechnic Inst. & St. Univ. Computer Science Department Blacksburg, Virginia 24061		8. CONTRACT OR GRANT NUMBER(s) AFOSR 79-0021
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NM Bolling AFB, Washington, DC 20332		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A2
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE September 1978
		13. NUMBER OF PAGES 27
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) APL Workspace Interchange, APL Workspace Transfer, Software Exchange		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Most suppliers of APL have not yet implemented the STAPL convention for transmitting workspaces from one installation to another. This report describes three workspace representations which may be used on a DECsystem-10 for this purpose. Two are partial implementations of the STAPL convention: one for level 2 of the convention, the other for level 3. The third representation is a terminal transcript file which is used as an input file. In addition, these representations may be used to reduce the disk storage required for APL workspaces on the DECsystem-10.		